Discrete Logarithm Variants of VSH

Arjen K. Lenstra[1,2] and Daniel Page[3] and Martijn Stam[1]

[1] EPFL / IC - LACAL, INJ3.33,
Station 14, CH-1015 Lausanne, Switzerland.
martijn.stam@epfl.ch
[2] Bell Laboratories.
[3] Dept. Computer Science, University of Bristol, Merchant Venturers Building,
Woodland Road, Bristol, BS8 1UB, United Kingdom.
page@cs.bris.ac.uk

Abstract. Recent attacks on standardised hash functions such as SHA1 have reawakened interest in design strategies based on techniques common in provable security. In presenting the VSH hash function, a design based on RSA-like modular exponentiation, the authors introduce VSH-DL, a design based on exponentiation in DLP-based groups. In this article we explore a variant of VSH-DL that is based on cyclotomic subgroups of finite fields; we show that one can trade-off performance against bandwidth by using known techniques in such groups. Further, we investigate a variant of VSH-DL based on elliptic curves and extract a tighter reduction to the underlying DLP in comparison to the original VSH-DL proposal.

Keywords. Hash Functions, Cyclotomic Subgroup, Collision Resistance.

1 Introduction

Hash function design Hash functions can be considered, together with block ciphers, to be the core primitives on which modern applied cryptography is based. The design of block ciphers is guided by a fairly mature and well understood background, see for example linear [14] and differential [3] cryptanalysis and the wide-trail design strategy of the AES [7]. In contrast, standardised hash functions such as SHA1 are constructed using somewhat ad-hoc techniques and they are essentially derived from the same family. This fact has, in part, contributed to a number of recent collision attacks against designs including SHA1 [20, 21].

Ideally, a hash function with output length n is a parameterised, deterministic function $\mathcal{H} : \{0,1\}^* \rightarrow \{0,1\}^n$ that takes an arbitrary length bitstring and maps it to a bitstring of length n. A good hash function satisfies several properties, the three most important of which are stated informally below.

1st-Preimage resistance Given a random image $x \in \{0,1\}^n$, it should take time $\approx 2^n$ to find $m \in \{0,1\}^*$ such that $\mathcal{H}(m) = x$.

2nd-Preimage resistance Given a 'random' $m \in \{0,1\}^*$, it should take time $\approx 2^n$ to find $m' \in \{0,1\}^*$ such that $\mathcal{H}(m) = \mathcal{H}(m')$ and $m \neq m'$.

Collision resistance It should take time $\approx 2^{n/2}$ to find $m, m' \in \{0,1\}^*$ such that $\mathcal{H}(m) = \mathcal{H}(m')$, yet $m \neq m'$.

Since generic black box attacks are known that find collisions in time $\approx 2^{n/2}$ or preimages in time $\approx 2^n$, the above requirements are very strong. In many scenarios it suffices to achieve a relaxed notion of collision resistance, in the sense that attackers who can invest only time 2^k cannot find collisions, where possibly the output length n is larger than $2k$. Thus, these hash functions might not strictly satisfy the standard security notion, even though collision resistance may provably be linked to a well studied hard problem, using the type of exact reduction also known from provable security. The tightness of the reduction and our belief or current understanding of the hardness of the underlying problem then lead to a parameter choice for which the resulting hash function has the desired collision resistance.

Hash functions based on modular exponentiation One of the first provably secure collision resistant hash functions is based on exponentiation modulo an RSA modulus, that is $\mathcal{H}(m) = x^m \bmod N$ where m is the message, N is the RSA modulus and x is some predefined value in $\mathbb{Z}_N^*$. If m and m' form a collision such that $\mathcal{H}(m) = \mathcal{H}(m')$, then $x^{m-m'} = 1 \bmod N$ which implies that $(m - m')$ is a multiple of the order of x. This order will necessarily be a divisor of $\phi(N)$ and if certain conditions hold, knowing any (nonzero) multiple of the order of x suffices to factor N in deterministic polynomial time. Note that there is no restriction on the length of m which means that there is no need for Merkle-Damgård [8, 15] type constructions.

This scheme was recently extended by Contini et al. [6] who essentially propose to use multi-exponentiation for the compression function instead of single exponentiation, thus obtaining an improvement in performance by processing more message bits at the same time. Let p_i be the i-th prime number, for $i = 1, \ldots, k$, where the product of the k primes should be smaller than the RSA modulus. A message m is then split up into l blocks M_i of equal length and the hash is computed as the multi-exponentiation $\mathcal{H}(m) = \prod_i p_i^{M_i} \bmod N$. An additional requirement is that the total bitlength of the message m is smaller than 2^k.

One of the disadvantages of VSH is the need for a secret RSA-modulus N. Someone who knows how to factor N can construct collisions easily. A side-effect of this trapdoor against collision resistance is that the modified Cramer-Shoup signature scheme [6] based on VSH does not provide non-repudiation as one might expect (cf. 'Creating Collisions' [6, p. 171]). Another disadvantage is the relatively large output length, namely the size of the RSA modulus. This means that to provide 80-bit security, one needs to use a hash function outputting approximately 1024 bits, rather than the desired 160 bits needed to thwart generic birthday attacks. To address these problems, Contini et al. mentioned the possibility of building VSH-DL, a hash function based on multi-exponentiation in DLP-based groups allowing short representations, such as elliptic curves or cyclotomic subgroups (allowing trace or torus-based methods). This design extends the corpus of previous work on DLP based hash functions, see [1, 2] for example.

Computation in finite field extensions The possibility to use finite fields with extension degree higher than one for public key cryptography has been known since the birth of public-key cryptography. However, for a long time nobody paid much attention to the subject since it was unclear whether the higher extension degree would offer any significant advantage over prime fields. It was not until Lenstra and Verheul showed the potential of working in a smaller subgroup of a larger field using their trace-based method called XTR [13] that interest increased.

Since then, Stam and Lenstra [19] showed how to efficiently work in the cyclotomic subgroup of a degree six extension field (provided that the characteristic satisfies a mild congruency relation), and Rubin and Silverberg [16] showed how to compress and decompress elements in this same subgroup using the theory of algebraic tori. The method of Rubin and Silverberg, called CEILIDH, differs from XTR in that compression is injective allowing full and exact decompression (in XTR conjugates are mapped to the same element). The downside of CEILIDH is that it is only a compression and decompression mechanism: it does not support direct computation on the compressed elements. Efficient arithmetic is still possible though, for instance by the method developed by Stam and Lenstra or the more involved hybrid methods by Granger et al. [10].

Main contributions Since methods known from the study of arithmetic and schemes using cyclotomic subgroups can provide computational efficiency and reduced bandwidth due to their compression properties, it is a natural question to ask to what extent they can be used to implement VSH-DL. To address this question, in this paper we investigate VSH-DL type schemes based on the cyclotomic subgroup of a sixth degree extension field and on elliptic curves.

Such schemes provide natural efficiency in terms of bandwidth which leads to a smaller hash-output without compromising security against collision attacks; through an experimental implementation we reason that this benefit is balanced against decreased performance compared to the original VSH-DL proposal. We do not make any claims about other security properties of our proposed hash functions, although it is easy to see that finding preimages is at least as hard as finding a collision. Thus it is possible to pick a (longer) output length of the hash function such that one also has the desired level of security against these two attacks. We believe that in many applications the level of security against collision attacks and preimage attacks can be set the same. Because our hash functions essentially depend on $n > 2k$, it is not recommended to truncate the output of the hash function (cf. [17]).

The paper is organised as follows. After our introduction of VSH in Section 2, we explore the possibility to base a hash function on a problem related to the discrete logarithm problem in the cyclotomic subgroup of a sixth degree extension field in Section 3, where we achieve a compression by a factor of three which represents a trade-off against decreased performance versus the original VSH-DL under a similar assumption. In Section 4 we discuss the possibility to use elliptic curves over prime fields. In that case the collision resistance of the hash function can be based directly on ECDLP. Finally we present some experimental results and analysis in Section 5 before concluding in Section 6.

2 VSH and VSH-DL

Contini et al. [6] define and analyse a hash function called Very Smooth Hash (VSH), which is a multi-exponentiation generalisation of the well known RSA-based hash. They write down the multi-exponentiation as a square-and-repeated-multiply algorithm, where they consider the processing of k bits (Step 5 in Algorithm 1 below) as a compression primitive and view the full compression function as repeated application of this compression primitive, which allows the computation of the hash function in a streaming fashion. Recall that for $i \in \mathbb{Z}_{>0}$, we let p_i is the i-th prime number.

Algorithm 1: VSH compression function [6].

Let N be an RSA modulus, let the block length k be the largest k for which $\prod_{i=1}^{k} p_i < N$. The VSH compression function $\mathcal{H}_{VSH} : \{0,1\}^{<2^k} \rightarrow \mathbb{Z}_N^*$ is defined as follows for an ℓ-bit message m consisting of bits $m_1, m_2, \ldots, m_\ell$, where $\ell < 2^k$.

1. [Initialise] Set $x_0 \leftarrow 1$, $\mathcal{L} \leftarrow \lceil \frac{\ell}{k} \rceil$ and $j \leftarrow 0$.
2. [Padding] Set $m_i \leftarrow 0$ for $\ell < i \leq \mathcal{L}k$.
3. [Merkle-Damgård Strengthening] Let $\ell = \sum_{i=1}^{k} \ell_i 2^{i-1}$ with $\ell_i \in \{0,1\}$ be the binary representation of the message length ℓ. Set $m_{\mathcal{L}k+i} \leftarrow \ell_i$ for $0 < i \leq k$.
4. [Finished] If $j = \mathcal{L} + 1$ terminate with output $x_{\mathcal{L}+1}$.
5. [Hash next block] Set $x_{j+1} \leftarrow x_j^2 \times \prod_{i=1}^{k} p_i^{m_{j \cdot k+i}} \pmod N$.
6. [Increase j] Increase j by one. Go back to Step 4.

It is not too hard to see that if we define $M_i = \sum_{j=0}^{\mathcal{L}} 2^{\mathcal{L}-j} m_{jk+i}$, taking into account the padding and the strengthening, then the hash is computed as the multi-exponentiation $H(m) = \prod_i p_i^{M_i} \bmod N$. In particular this means that one might be able to achieve some speedups by using techniques known from the theory of addition chains.

Contini et al. mention precomputing products of primes: indeed, if k primes (or bases) are given and a small positive integer b divides k, we can partition the bases in k/b sets of b primes each and for each set precompute all 2^b products of the different primes in that set. During the actual hashing bits are processed in chunks of b bits so that only k/b multiplications will be needed to process k bits of message (this is essentially a simplified version of Pippenger's algorithm). Contini et al. observe that instead of using precomputed products of primes for chunks of bits, one can also use fresh primes instead. Although this leads to a different hash function, called Fast VSH, it is based on the same hardness assumption as standard VSH but, as the name suggests, considerably faster (also compared to VSH with precomputation). A full description can be found in Appendix A.

The collision resistance of VSH can be reduced to the VSSR problem.

Definition 2 *(VSSR: Very Smooth number nontrivial modular Square Root [6, Def. 3]) Let N be the product of two unknown primes of approximately the same size and let $k \leq (\log N)^c$. VSSR is the following problem: Given N, find $x \in \mathbb{Z}_N^*$ such that $x^2 = \prod_{i=1}^{k} p_i^{e_i}$ and at least one of $e_1, \ldots, e_k$ is odd.*

Contini et al. note that, given the existing known factoring algorithms, it seems as hard to solve the VSSR problem as it is to factor N (though they base their analysis on a

more conservative relation). They also define a discrete log analogue to VSSR leading to VSH-DL. An important advantage of VSH-DL over VSH is the lack of a trapdoor.

Definition 3 *(VSDL: Very Smooth number Discrete Log [6, Def. 4]) Let p, q be primes with $p = 2q + 1$ and let $k \leq (\log p)^c$. VSDL is the following problem: given p, find integers $e_1, e_2, \ldots, e_k$ such that $2^{e_1} \equiv \prod_{i=2}^{k} p_i^{e_i} \mod p$ with $|e_i| < q$ for $i = 1, 2, \ldots, k$, and at least one of $e_1, e_2, \ldots, e_k$ is non-zero (where p_i is to be understood to be the i-th prime number).*

Algorithm 4: VSH-DL compression function.

Let p be an S-bit prime of form $2q + 1$ for prime q, let k be a fixed integer length, typically $k \approx S/\log S$. The VSH-DL compression function $\mathcal{H}_{DL} : \{0,1\}^{<(S-2)k} \rightarrow \mathbb{Z}_p^*$ is defined as follows for an ℓ-bit message m consisting of bits $m_1, m_2, \ldots, m_\ell$, with $\ell < (S-2)k$.

1. [Initialise] Set $x_0 \leftarrow 1$, $\mathcal{L} \leftarrow \lceil \frac{\ell}{k} \rceil$ and $j \leftarrow 0$.
2. [Padding] Set $m_i \leftarrow 0$ for $\ell < i \leq \mathcal{L}k$.
3. [Merkle-Damgård Strengthening] Let $\ell = \sum_{i=1}^{k} \ell_i 2^{i-1}$ with $\ell_i \in \{0,1\}$ be the binary representation of the message length ℓ. Set $m_{\mathcal{L}k+i} \leftarrow \ell_i$ for $0 < i \leq k$.
4. [Finished] If $j = \mathcal{L} + 1$ terminate with output $x_{\mathcal{L}+1}$.
5. [Hash next block] Set $x_{j+1} \leftarrow x_j^2 \times \prod_{i=1}^{k} p_i^{m_{j \cdot k+i}} \pmod p$.
6. [Increase j] Increase j by one. Go back to Step 4.

3 A Cyclotomic Subgroup Variant of VSH-DL

We begin with a brief overview of the mathematics underlying CEILIDH and XTR. This overview is specifically tailored to our needs, for a more general introduction see [9] and the references contained therein.

Let p be a prime and let $\mathbb{F}_p$ denote a finite field of order p and $\mathbb{F}_{p^6}$ a sixth degree extension thereof. The multiplicative group $\mathbb{F}_{p^6}^*$ is cyclic of order $p^6 - 1$, which factors as $(p^2-p+1)(p^2+p+1)(p+1)(p-1)$. Let G be the unique subgroup of order p^2-p+1 in $\mathbb{F}_{p^6}^*$. We call G the cyclotomic subgroup of $\mathbb{F}_{p^6}^*$. Alternatively, it can be regarded as a specific algebraic torus of dimension 2 over $\mathbb{F}_p$. It is argued [12] that the computational complexity of the discrete logarithm problem in $\mathbb{F}_{p^6}^*$ resides in this subgroup G of order $p^2 - p + 1$, since the subgroups of order dividing $(p^2 + p + 1)(p + 1)(p - 1)$ can be efficiently embedded in proper subfields of $\mathbb{F}_{p^6}^*$, thus allowing to run a sub-exponential algorithm in the smaller field.

Using a standard representation in $\mathbb{F}_{p^6}$ consumes $\approx 6 \log p$ bits which seems wasteful given that there are only $\approx p^2$ elements in G. This problem can be solved using either XTR [13] or CEILIDH [16]. With XTR, the trace map

$$\mathrm{Tr} : \mathbb{F}_{p^6} \rightarrow \mathbb{F}_{p^2} : x \rightarrow x^{p^4} + x^{p^2} + x$$

is used to compress an element in G to an element in $\mathbb{F}_{p^2}$. This map is not injective; since conjugates over $\mathbb{F}_{p^2}$ map to the same value in $\mathbb{F}_{p^2}$ it is essentially 3-to-1. One of

the significant advantages of XTR is that it is possible to work directly with compressed elements when performing an exponentiation. Unfortunately, this method does not generalise very well to multi-exponentiation on more than two bases which makes XTR unsuitable for direct use in a VSH-DL variant.

CEILIDH is an alternative to XTR that offers only compression; that is, one cannot compute directly with compressed elements. Formally, CEILIDH is a bijection between $\mathsf{G}\backslash\{a\}$ and $(\mathbb{F}_p)^2\backslash V(f)$, where a is some particular element of G and $V(f)$ is a well-defined subset of $(\mathbb{F}_p)^2$ (the notation $V(f)$ stems from the fact that it is a variety defined by a single polynomial). It is straightforward to extend CEILIDH into an injection from G to $(\mathbb{F}_p)^2$. Clearly, given any collision-resistant hash function, applying an injection to the result is not going to reduce its collision resistance.

One of the advantages of VSH-DL as described by Contini et al. is its use of small elements p_i. When we work directly with elements in G, there do not seem to be elements that are as efficient to work with. Since each group element is represented by six base field elements and the order of the group is $p^2 - p + 1$ one can prescribe at most two base field elements to be small, the other four will follow and be of normal size. However, an alternative presents itself by not working with elements in G, but rather elements in the full field $\mathbb{F}_{p^6}$ and only map to G at the very end through powering by $(p^3 - 1)(p + 1)$, i.e. the cofactor of G in $\mathbb{F}_{p^6}^*$). Thus our hash function will be

$$\mathrm{Ceilidh}((\prod_i h_i^{M_i})^{(p^3-1)(p+1)})$$

for suitably chosen $h_i \in \mathbb{F}_{p^6}^*$.

In the following we will motivate this approach by showing that, when the elements h_i are chosen uniformly at random and then somehow replaced by a smaller sibling in the same coset (under exponentiation with the cofactor of G), the hash function is collision-resistant if the discrete logarithm problem in (a subgroup of) G is hard. Thus, contrary to the original VSH-DL, we reduce from a standard discrete logarithm assumption. Our proof uses a slightly more general notation than as above. We note that the order of G itself need not be prime, although to provide collision resistance the size of G's largest DLP-hard subgroup of prime order will be relevant for determining the maximal allowable message length. Our reduction is similar to that of Bellare and Micciancio [2], but tighter because we reduce from the DLP in a prime order subgroup.

Definition 5 *(DLP) Let G_q be a finite cyclic group of known prime order q and with generator f. The discrete logarithm problem in G_q is to find, given y drawn uniformly at random from G_q, the unique value $0 \leq x < q$ such that $y = f^x$.*

Definition 6 *(k-modified DLP) Let H be a finite cyclic group with generator h. Let G be a subgroup of H with generator $g = h^{|\mathsf{H}|/|\mathsf{G}|}$. Let $\psi : \mathsf{H} \to \mathsf{H}$ be a map such that $\psi(h_i)^{|\mathsf{H}|/|\mathsf{G}|} = (h_i)^{|\mathsf{H}|/|\mathsf{G}|}$ for all $h_i \in \mathsf{H}$. The k-modified discrete logarithm problem for $(\mathsf{H}, \mathsf{G}, \psi)$ is to find, given h_i drawn uniformly at random from H for $i = 1, \ldots, k$, a nonzero solution $(e_1, \ldots, e_k) \in [0, q)^k$ of*

$$(\prod_{i=1}^{k} \psi(h_i)^{e_i})^{|\mathsf{H}|/|\mathsf{G}|} = 1$$

Theorem 7 *Assume q divides $|\mathsf{G}|$, but q^2 does not divide $|\mathsf{H}|$ and that $k > 1$. An attacker that solves the k-modified discrete logarithm problem for $(\mathsf{H}, \mathsf{G}, \psi)$ in time t with probability ϵ can be used to solve the discrete logarithm problem in $\mathsf{G}_q \subseteq \mathsf{G}$ in time $t + t'$ with probability $\epsilon - 1/q$, where t' is essentially the time to perform a k-fold double exponentiation in H.*

Proof:

Given $y \in \mathsf{G}_q$, we need to find x such that $f^x = y$ with the help of an attacker that solves the k-modified discrete logarithm problem. Let $h_1 = y^{b_i} h^{a_i}$ for $i = 1, \ldots, k$, where the a_i and b_i are drawn uniform at random from $[0, |\mathsf{H}|)$. As a result the h_i are distributed uniformly as expected by the k-modified DLP attacker, so on input of these h_i the attacker will, with probability ϵ, return $(e_1, \ldots, e_k)$ such that

$$(\prod_{i=1}^{k} \psi(y^{b_i} h^{a_i})^{e_i})^{|\mathsf{H}|/|\mathsf{G}|} = 1$$

hence

$$g^{\sum_{i=1}^{k} a_i e_i} = y^{-(|\mathsf{H}|/|\mathsf{G}|) \sum_{i=1}^{k} b_i e_i} .$$

Note that $y = f^x$ for some (yet unkown) value of $0 \leq x < q$ and that, w.l.o.g., $f = g^{|\mathsf{G}|/q}$. Since g is a generator of $|\mathsf{G}|$, this implies that

$$\sum_{i=1}^{k} a_i e_i = -x(|\mathsf{H}|/q) \sum_{i=1}^{k} b_i e_i \bmod |G|$$

and because q divides $|G|$ also

$$\sum_{i=1}^{k} a_i e_i = -x(|\mathsf{H}|/q) \sum_{i=1}^{k} b_i e_i \bmod q .$$

Now we can compute x if $(|\mathsf{H}|/q) \sum_{i=1}^{k} b_i e_i \neq 0 \bmod q$. Since q^2 does not divide $|\mathsf{H}|$, we know that $|\mathsf{H}|/q$ is invertible modulo q, so we need to show that $\sum_{i=1}^{k} b_i e_i \neq 0 \bmod q$ with probability at most $1/q$.

Because, given any $h_i \in \mathsf{H}$ and $0 \leq b_i < q$, there is exactly one a_i such that $h_i = h^{a_i} y^{b_i}$, it follows that the adversary given the h_i has no Shannon information on b_i when announcing the e_i. Consequently, unless all e_i are congruent to 0 modulo q (which is not allowed by the restrictions on the e_i), $\sum_{i=1}^{k} b_i e_i$ is uniformly randomly distributed modulo q, so the probability of it being $0 \bmod q$ is $1/q$.

Q.E.D.

Algorithm 8: Modified VSH-DL compression function.

Let H be a finite cyclic group of known, factored order and generator h. Let G be a subgroup of H with generator $g = h^{|\mathsf{H}|/|\mathsf{G}|}$. Let q be a prime dividing $|\mathsf{G}|$ but not $|\mathsf{H}|/q$. Let $\psi : \mathsf{H} \to \mathsf{H}$ be a map such that $\psi(h_i)^{|\mathsf{H}|/|\mathsf{G}|} = (h_i)^{|\mathsf{H}|/|\mathsf{G}|}$ for all $h_i \in \mathsf{H}$.

Let $\mathrm{Compress} : \mathsf{G} \to R$ be an efficiently computable injection. Let k be a fixed integer length such that $(\lfloor \log_2 q \rfloor - 1)k < 2^k$. The modified VSH-DL compression function $\mathcal{H}_{MDL} : \{0,1\}^{\leq (\lfloor \log_2 q \rfloor - 1)k} \to R$ is defined for an ℓ-bit message m consisting of bits $m_1, m_2, \ldots, m_\ell$ with $\ell \leq (\lfloor \log_2 q \rfloor - 1)k$ as follows.

1. [Initialise] Set $x_0 \leftarrow 1$, $\mathcal{L} \leftarrow \lceil \frac{\ell}{k} \rceil$ and $j \leftarrow 0$.
2. [Padding] Set $m_i \leftarrow 0$ for $\ell < i \leq \mathcal{L}k$.
3. [Merkle-Damgård Strengthening] Let $\ell = \sum_{i=1}^{k} \ell_i 2^{i-1}$ with $\ell_i \in \{0,1\}$ be the binary representation of the message length ℓ. Set $m_{\mathcal{L}k+i} \leftarrow \ell_i$ for $0 < i \leq k$.
4. [Finished?] If $j = \mathcal{L} + 1$ terminate with output $\mathrm{Compress}(x_{\mathcal{L}}^{|\mathsf{H}|/|\mathsf{G}|})$.
5. [Hash next block] Set $x_{j+1} \leftarrow x_j^2 \times \prod_{i=1}^{k} \psi(h_i)^{m_{j \cdot k+i}}$.
6. [Increase j] Increase j by one. Go back to Step 4.

For completeness, we note that the collision resistance of the hash function above can be related to k-modified DLP by observing that the (implicit) map turning messages into the relevant exponents for h_i is injective on its domain (this is the reason for the length restrictions on the message).

As mentioned before, we will instantiate Algorithm 8 with $\mathsf{H} = \mathbb{F}_{p^6}^*$ and G a subgroup of order $p^2 - p + 1$ which has cofactor $(p^3 - 1)(p + 1)$. Moreover for $\mathrm{Compress}$ we will substitute $\mathrm{Ceilidh}$. For efficient field arithmetic we restrict ourselves to $p \equiv 2 \bmod 9$. In this case p will generate $\mathbb{Z}_9^*$ and $\Phi_9(x) = x^6 + x^3 + 1$ is irreducible in $\mathbb{F}_p$. Hence if γ is a root of $\Phi_9(x)$ (i.e., a ninth root of unity), then $(\gamma, \gamma^2, \gamma^3, \gamma^4, \gamma^5, \gamma^6)$ is a basis for the extension field $\mathbb{F}_{p^6} = \mathbb{F}_p[\gamma]$. The arithmetic based on this extension also lies at the basis of the fast implementation [10] of XTR and CEILIDH.

Let $a = \sum_{j=0}^{5} a_j \gamma^{j+1} \in \mathbb{F}_{p^6}$. We are interested in finding a small representation $\psi(a)$ of a such that multiplication of an arbitrary field element by $\psi(a)$ will be relatively cheap. We do this implicitly. Instead of giving a straightforward definition of ψ we show how to sample efficiently and (almost) uniformly from $\psi(\mathbb{F}_{p^6}^*)$. This is done by ensuring that $\psi(\mathbb{F}_{p^6}^*)^{(p^3-1)(p+1)}$ gives rise to the (almost) uniform distribution over G.

We sample from $\psi(\mathbb{F}_{p^6}^*)$ as follows. Draw a_0 and a_5 uniformly and independently at random from $\mathbb{F}_p$. Let $\psi(a) = a_0\gamma + \gamma^2 + a_5\gamma^6$. That this works follows from the following observation:

Lemma 9 *The map $\psi' : \mathbb{F}_p^2 \to \mathsf{G}$ defined by $\psi'(a_0, a_5) = (a_0\gamma + \gamma^2 + a_5\gamma^6)^{(p^3-1)(p+1)}$ has a range of at least $p^2/3$ elements.*

Proof: To prove the statement we will upper bound the number of collisions, that is, sets of distinct pairs (a_0, a_5) and (b_0, b_5) such that $\psi'(a_0, a_5) = \psi'(b_0, b_5)$. Equivalently, we are counting the sets a and b of prescribed form $a = a_0\gamma + \gamma^2 + a_5\gamma^6$ and $b = b_0\gamma + \gamma^2 + b_5\gamma^6$ such that $a^{(p^3-1)(p+1)} = b^{(p^3-1)(p+1)}$. The latter equation can be rewritten as $(a^{p^3}b)^{p+1} - (ab^{p^3})^{p+1}) = 0$. This can be computed algebraically, giving rise to initially six equations (one for each coordinate), but that can be simplified to

$$(a_0 - b_0)(1 + a_0b_0 + a_5b_5) = 0$$
$$(a_5 - b_5 + 2(a_0 - b_0) + a_0b_5 - a_5b_0)(1 + a_0b_0 + a_5b_5) = 0$$

plus a third condition. Simultaneously satisfying the above two equations can only be done if either $(a_0, a_5) = (b_0, b_5)$ or if $(1+a_0b_0+a_5b_5) = 0$. Unless $(b_0, b_5) = (0, 0)$ the latter solution allows us substitution of either a_0 or a_5 in the third and final equation to be satisfied (and note that only $(0, 0)$ is in the preimage of $\psi'(0, 0)$). This third equation will then yield a quadratic equation in either a_0 or a_5 that is only degenerate (i.e. equal to zero) if both $b_0 = 0$ and $b_5 = 0$. Since a quadratic equation over a finite field has at most two solutions, we have shown that any preimage of ψ' has cardinality at most three, from which the claim follows. *Q.E.D.*

One can improve performance considerably by picking small a_0 and a_5. The caveat is that the security is no longer directly related to a clean DLP assumption. Our choice will be to set $(a_5)_i$ equal to 1 and let $(a_0)_i$ depend on i, i.e. simply range through a number of a_0 values that are easy to multiply with. In particular, we use $(a_0)_i = i + 1$ for a given i.

For completeness we note that the preimage under ψ' of $1 \in G$ in this case is restricted to $a_0 = 0$ or $a_0 = 1$, which can be seen by using that $a^{(p^3-1)(p+1)} = 1$ is equivalent to $a^{(p^2+p+1)(p+1)} \in \mathbb{F}_p^*$. Moreover, if $a = a_0\gamma + \gamma^2 + \gamma^6$ and $b = b_0\gamma + \gamma^2 + \gamma^6$, then $a^{(p^3-1)(p+1)} = b^{(p^3-1)(p+1)}$ iff $a = b$ or a, b are in the preimage of 1.

Thus we are insured that as long as we pick the a_0 distinct and unequal to 0 or 1 our system is not obviously flawed and there is no reason to assume that the choice of our $\psi(h_i)$ is weak. The resulting hash function can be proven secure assuming that solving the following, admittedly tailor-made, problem is hard:

Definition 10 *(Small Element DLP) Let p be a prime congruent to $2 \bmod 9$ such that $p^2 - p + 1$ has at least one big prime factor q. Let γ be a ninth-root of unity and let $h_i = (i + 1)\gamma + \gamma^2 + \gamma^6 \in \mathbb{F}_{p^6}^*$ for $i = 1, \ldots, k$. The small element discrete logarithm problem is to find, given p, a nonzero solution $(e_1, \ldots, e_k) \in [0, q)^k$ of*

$$(\prod_{i=1}^{k} \psi(h_i)^{e_i})^{(p^3-1)(p+1)} = 1 \, .$$

4 An Elliptic Curve Variant of VSH-DL

Since their introduction to cryptography by Koblitz and Miller, elliptic curves have become ever more popular as a replacement for finite fields to base DLP-based schemes on. This is mainly due to the fact that there is no known algorithm to solve the DLP on a general elliptic curve faster than the generic Pollard-ρ method. This allows one to use curves with group sizes quadratic in the security one wishes to offer.

An immediate result of this is that to obtain 2^k bit security against collision-finding, one can actually use a hash function based on VSH-DL that outputs just over $2k$ bits (by using affine representation and standard point compression, where the Y-coordinate is replaced by a single bit to resolve any square root ambiguity). Moreover, the computations are relatively fast. In this article we concentrate on curves over prime fields, though similar results are expected to hold for curves over binary or ternary fields.

An elliptic curve over a prime field $\mathbb{F}_p$ with $p > 3$ can be represented using the short Weierstrass form

$$Y^2 = X^3 + a_4X + a_6$$

where it is common to use $a_4 = -3$, for example in the NIST standard curves, in order to extract some performance benefits. The set of points $(X, Y) \in \mathbb{F}_p^2$ satisfying the equation above together with a point at infinity form an abelian group under the addition operation also known as the chord-tangent process. The point at infinity serves as group identity and the negation of a point (X, Y) is $(X, -Y)$.

Optimising elliptic curve arithmetic has been the focus of a large number of articles. Excellent overviews are given by Brown et al.[5] and Hankerson et al.[11]. One of the most efficient methods is the use of mixed coordinates, where the fixed multiplicands are kept in affine representation but where computations are done using the Jacobian representation. Point doubling with the Jacobian representation costs 4 field multiplications and 4 field squarings. Adding an affinely represented point to a point in Jacobian representation costs 8 field multiplications and 3 field squarings. The result again is in the Jacobian representation.

One could consider the use of small points P_i. However, it seems that only one of the coordinates of P_i can be picked small, since the other coordinate typically follows from the curve equation (indeed, if $a_4 = -3$ and a_6 has full size, it is impossible for both the X and the Y-coordinate of a point on the curve to be small). It is easy to see from [11, Algorithm 3.22] that both coordinates are used only once during the point addition, so picking small P_i's will reduce the cost of a point addition by at most one field multiplication.

A possible solution to this problem is the use of a Montgomery representation. Recently Brown [4] showed how to perform a multi-exponentiation efficiently in this setting.

5 Experimental Results

Strategy The crucial operation in the implementation of VSH is

$$x_{j+1} \leftarrow x_j^2 \prod_{i=1}^{k} p_i^{m_{j \cdot k+1}} \pmod{n}.$$

This product can be computed in several ways by reshuffling the order in which the multiplications take place. One option is to start with x_j^2 and then multiply by the relevant p_i terms one by one, reducing each time. The option recommended by Contini et al. is to first compute the product

$$P = \prod_{i=1}^{k} p_i^{m_{j \cdot k+1}}$$

and then multiply by x_j^2. Due to the choice of parameters, the product P will be smaller than the modulus N when computed over the integers, so modular reductions are not

	slow	fast $b = 2$	fast $b = 4$	fast $b = 8$
SHA1-160	26.29 (or 16.89 with SIMD)			
VSH	632.36	622.11	370.37	277.60
VSH-DL-$\mathcal{A}$	715.71	676.04	382.68	274.75
VSH-DL-$\mathcal{B}$	5507.54	6244.37	3126.34	1338.18
VSH-DL-$\mathcal{C}$	16080.26	11777.89	7542.05	4105.31

Table 1. A comparison between the original VSH design and three variants of VSH-DL. Results are given in clock cycles per byte of input.

necessary. VSH relies on this feature of the p_i, or small element values to enable the construction of high performance implementations. It also acts as the bottleneck for our compressed VSH-DL variants. Specifically, it is much harder to reason about how one would delay reductions in either the cyclotomic subgroup or elliptic curve cases: although one can attempt to construct some notion of small elements, it is difficult to imagine how performing computation with such elements will be as efficient as in the original VSH case.

Results In order to evaluate the relative performance characteristics of our VSH-DL designs versus the original proposal by Contini et al. [6], we produced some experimental results using an implementation in C. Our platform for these experiments was a 2.8Ghz Intel Pentium 4; we used GCC $4.0.1$ to compile our implementation which relied on NTL [18] for the underlying arithmetic. We produced an implementation of the original VSH scheme and three variants of the VSH-DL scheme as detailed below:

VSH The original VSH scheme operates modulo an RSA number $N = pq$ for primes p and q. The parameters p and q were selected such that $\log_2(N) = 1024$.

VSH-DL-$\mathcal{A}$ The first variant represents the original VSH-DL scheme of Contini et al. by working in the group of integers modulo a prime p such that $p = 2q + 1$ for some prime q. The parameters p and q were selected such that $\log_2(p) = 1152$.

VSH-DL-$\mathcal{B}$ The second variant represents the cyclotomic subgroup based VSH-DL design as described in Section 3. It works in a group G which is a subgroup of $\mathbb{F}^*_{p^6}$. The parameter p was selected such that $\log_2(p) = 192$. In this implementation we were careful to use delayed reduction techniques to improve arithmetic in G, and to construct a dedicated multiplication function for multiplication by small elements which have a special, sparse form.

VSH-DL-$\mathcal{C}$ The final variant represents the elliptic curve based VSH-DL design as described in Section 4. It works in a prime subgroup of a curve $E(\mathbb{F}_p)$. The parameter p was selected such that $\log_2(p) = 192$. We considered only random curves of the form

$$E : Y^2 = X^3 - 3X + a_6$$

such that special reduction techniques such as those for Mersenne primes were not available; one might expect an incremental improvement in performance by using

such a parameterisation. We used Jacobian projective coordinates and a mixed-addition strategy as is common in many point multiplication methods. The form of arithmetic on the curve meant that a dedicated point addition function for addition of small elements did not give any significant benefit.

For each variant, our parameter selection is such that the security of the resulting hash functions is roughly equal. However, we make no attempt to to select values of k that are sensible for the different variants; we use $k = 131$ in all cases. The results in Table 5 detail the performance of our variants; the figures quoted are clock cycles per byte of input measured using the `rdtsc` instruction. They do not include the cost of initialising the hash function, for example pre-computation of any tables of small elements. We include results, following the nomenclature of Contini et al., for both slow and fast versions: the fast version blocks the input and uses some pre-computation to improve performance.

Analysis Even considering incremental performance improvements by using, for example, Mersenne primes in the elliptic curve case, our VSH-DL variants perform at least an order or magnitude worse than either the original VSH or VSH-DL proposals by Contini et al. In this respect, their worth in terms of pure performance is untenable. This is exacerbated when one considers that hardware acceleration for modular multiplication as used in VSH is commonplace as a result of use in RSA; one might expect significant performance improvements in a practical setting as a result.

However, one area of advantage which our cyclotomic subgroup variant of VSH-DL gives is memory footprint. That is, the computation of small field elements is essentially free in comparison with the computation of the small primes used in VSH. For the figures in [6][Section 5] one can see that VSH pays a hefty price in terms of memory real-estate to achieve the levels of performance indicated in our results. Although the performance is lower, our cyclotomic subgroup variant of VSH-DL requires far less memory.

A more subtle issue is the selection of the k parameter. By increasing k, which roughly speaking is the block size of the hash function, one can decrease the number of squaring operations in the compression function; the number of multiplications stays the same. The choice of k for VSH is motivated by the need to avoid modular reductions in computing the product P from above. In our schemes we have already highlighted the fact that it is not easy to avoid such reductions; as such we can be more flexible in our choice of k.

6 Conclusion

In this article we have examined in depth the possibility to base VSH-DL on either cyclotomic subgroups of finite fields of extension degree six or on elliptic curves of large prime characteristic. We concluded that for cyclotomic subgroups using CEILIDH we can get a hash function that is about an order of magnitude slower than the original VSH-DL proposal, secure under a slightly different assumption (due to an inevitable redefining of what constitutes a small element), but has a compression factor that is

three times as high. Using elliptic curves, we derive a hash function that is significantly slower than the original VSH-DL proposal, but whose compression factor is six times as high and its security can be directly linked to that of the standard ECDLP. In both cases the poor performance is balanced by a potential saving in terms of memory footprint.

We reiterate that our main concern was provable collision resistance under a discrete logarithm-like assumption. Like VSH and VSH-DL on which our constructions are based, our scheme provides only collision resistance and may not be suitable to replace a random oracle in all situations.

References

1. M. Bellare, O. Goldreich and S. Goldwasser. Incremental Cryptography: The Case of Hashing and Signing. In *Advances in Cryptology (CRYPTO)*, Springer-Verlag LNCS 839, 216–233, 1994.
2. M. Bellare and, D. Micciancio. A New Paradigm for Collision-Free Hashing: Incrementality at Reduced Cost. In *Advances in Cryptology (EUROCRYPT)*, Springer-Verlag LNCS 1233, 163–192, 1997.
3. E. Biham and A. Shamir. Differential Cryptanalysis of DES-like Cryptosystems. In *Advances in Cryptology (CRYPTO)*, Springer-Verlag LNCS 537, 2–21, 1990.
4. D. R. L. Brown. Multi-Dimensional Montgomery Ladders for Elliptic Curves. IACR eprint, 2006/220, 2006.
5. M. Brown, D. Hankerson, J. López, and A. Menezes. Software implementation of the NIST elliptic curves over prime fields. In D. Naccache, editor, *CT-RSA'01*, volume 2020 of *Lecture Notes in Computer Science*, pages 250–265. Springer-Verlag, 2001.
6. S. Contini, A. K. Lenstra, and R. Steinfeld. VSH, an efficient and provable collision resistant hash function. In S. Vaudenay, editor, *Advances in Cryptography—Euro'06*, volume 4004 of *Lecture Notes in Computer Science*, pages 165–182. Springer-Verlag, 2006.
7. J. Daemen and V. Rijmen. *The Design of Rijndael*. Springer-Verlag, 2002.
8. I.B. Damgård. Collision Free Hash Functions and Public Key Signature Schemes. In *Advances in Cryptology (EUROCRYPT)*, Springer-Verlag LNCS 304, 203–216, 1987.
9. R. Granger. *On Small Degree Extension Fields in Cryptology*. PhD thesis, University of Bristol, 2005.
10. R. Granger, D. Page, and M. Stam. A comparison of ceilidh and xtr. In D. Buell, editor, *ANTS-VI*, volume 3076 of *Lecture Notes in Computer Science*, pages 235–249. Springer-Verlag, 2004.
11. D. Hankerson, A. Menezes, and S. Vanstone. *Guide to Elliptic Curve Cryptography*. Springer-Verlag, 2004.
12. A. K. Lenstra. Using cyclotomic polynomials to construct efficient discrete logarithm cryptosystems over finite fields. In V. Varadharajan, J. Pieprzyk, and Y. Mu, editors, *ACISP'97*, volume 1270 of *Lecture Notes in Computer Science*, pages 127–138. Springer-Verlag, 1997.
13. A. K. Lenstra and E. R. Verheul. The XTR public key system. In M. Bellare, editor, *Advances in Cryptography—Crypto'00*, volume 1880 of *Lecture Notes in Computer Science*, pages 1–19. Springer-Verlag, 2000.
14. M. Matsui Linear Cryptanalysis Method for DES Cipher. In *Advances in Cryptology (EUROCRYPT)*, Springer-Verlag LNCS 765, 386–397, 1993.
15. R.C. Merkle. A Fast Software One-way Hash Function. *Journal of Cryptology*, **3**, 43–58, 1990.
16. K. Rubin and A. Silverberg. Torus-based cryptography. Technical Report 39, IACR's ePrint Archive, 2003.

17. M.-J. O. Saarinen. Security of VSH in the real world. Technical Report 103, IACR's ePrint Archive, 2006.
18. V. Shoup. NTL: A Library for doing Number Theory. Available from: http://www.shoup.net/ntl/
19. M. Stam and A. K. Lenstra. Efficient subgroup exponentiation in quadratic and sixth degree extensions. In J. Burton S. Kaliski, Ç. Koç, and C. Paar, editors, *CHES'02*, volume 2523 of *Lecture Notes in Computer Science*, pages 318–332. Springer-Verlag, 2003.
20. X. Wang, H. Yu, and Y. L. Yin. Efficient Collision Search Attacks on SHA-0. In *Advances in Cryptology (CRYPTO)*, Springer-Verlag LNCS 3621, 1–16, 2005.
21. X. Wang, Y. Yin, H. Yu. Finding Collisions in the Full SHA-1. In *Advances in Cryptology (CRYPTO)*, Springer-Verlag LNCS 3621, 7–36, 2005.

A Fast VSH

Recall that for $i \in \mathbb{Z}_{>0}$, we let p_i be the i-th prime number. Let N be an RSA modulus, let k be the block length and let b be the chunking factor. To avoid intermediate reductions, one should ensure that $\prod_{i=1}^{k} p_{(2^b-1)i} < N$. Note that the Merkle-Damgård strengthening listed below might allow collisions on messages of length greater than 2^{bk}, but for reasonable parameter choices of b and k this will not be an issue.

Algorithm 11: VSH compression function [6].

The VSH compression function $\mathcal{H}_{VSH} : \{0,1\}^* \rightarrow \mathbb{Z}_N^*$ is defined as follows for an ℓ-bit message m consisting of bits $m_1, m_2, \ldots, m_\ell$.

1. [Padding] Set $\mathcal{L} \leftarrow \lceil \frac{\ell}{bk} \rceil$ and $m_i \leftarrow 0$ for $\ell < i \leq \mathcal{L}bk$.
2. [Radix Conversion] Set $M_i = \sum_{j=0}^{b-1} m_{b(i-1)+j+1} 2^j$ for $0 < i \leq k$.
3. [Merkle-Damgård Strengthening] Let $\ell = \sum_{i=1}^{k} \ell_i 2^{(i-1)b}$ with $\ell_i \in \{0, 2^b - 1\}$ be the 2^b-ary representation of the message length ℓ. Set $M_{\mathcal{L}k+i} \leftarrow \ell_i$ for $0 < i \leq k$.
4. [Initialise Loop] Set $x_0 \leftarrow 1$ and $j \leftarrow 0$.
5. [Padding] Set $m_i \leftarrow 0$ for $\ell < i \leq \mathcal{L}bk$.
6. [Finished] If $j = \mathcal{L} + 1$ terminate with output $x_{\mathcal{L}+1}$.
7. [Prepare product] Set $P \leftarrow \prod_{i=1}^{k} p_{M_i+(i-1)(2^b-1)}$ skipping those i for which $M - i = 0$.
8. [Hash next block] Set $x_{j+1} \leftarrow x_j^2 \times P \pmod N$.
9. [Increase j] Increase j by one. Go back to Step 6.